STRLEN

Beware of use with strings that are not null terminated.

Sean Barnum, Cigital, Inc. [vita[1]]

2007-04-23

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 5796 bytes

Attack Category	• Malicious Input • Denial of Service
Vulnerability Category	• Buffer Overflow • Unconditional
Software Context	• String Management
Location	
Description	The strlen() function can be associated with problems if a string is not null terminated or if it is used in a way that causes a null terminator to be lost. The strlen() function finds the length of the given string (represented as a character array). It iterates through the characters in the string, stopping when it finds the null terminator. The problem does not lie in the strlen() function itself but in how it is generally used. If an attacker is able to generate a string in an application that is not null terminated, strlen() will return an incorrect length. If this length is used to perform tasks such as memory allocation and string parsing, other security problems can occur. The use of strlen() can easily cause off-by-one errors, since copying a string of length strlen(buf) actually requires copying strlen(buf)+1 characters (including the null terminator). The resulting unterminated strings can cause subsequent problems when functions such as strlen() are run on them.
APIs	**Function Name** / **Comments**: _mbslen; _tcslen; lstrlen; strlen; wcslen
Method of Attack	In some cases, an attacker can place a string in a buffer that is exactly the length of the allocated

APIs:

Function Name	Comments
_mbslen	
_tcslen	
lstrlen	
strlen	
wcslen	

1. http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html (Barnum, Sean)

	buffer. This leaves no space for the terminating null character, effectively connecting the string with whatever is next in memory. Note that this is not a problem with the strlen() function itself but with the function used to populate the buffer. However, as a result, a call to strlen() might run off the end of the string and cause an access violation, thus terminating the application. There are also many ways in which improper use of strlen() can cause buffer overflows and other security problems.
Exception Criteria	When the string input to the strlen() function is defined by the programmer and is not modified during program execution.
Solutions	see below

Solution Applicability	**Solution Description**	**Solution Efficacy**
All uses of strlen() should be checked to ensure safe usage.	All buffers containing strings input by the user should be explicitly terminated with \0 before a call to strlen(). If the buffer is statically allocated, buffer[sizeof(buffer)-1] should be set to \0 before a call to strlen(). This way, even if the actual string is not null terminated, strlen() will not read past the buffer. If the buffer is dynamically allocated, then string objects that store the length of the string should be used instead of character arrays. In this case, calling strlen() should be unnecessary. Note that strlen() itself	Typically effective, but there are many ways of creating unsafe string management code.

	does not cause major security problems. It simply increases the chances of security problems in other string handling functions that use the output of strlen().	
When tempted to use strlen()	Use strlen_s() or StringCchLength()	

Signature Details	size_t strlen(const char *string) size_t wcslen(const wchar_t *string) size_t _mbslen(const unsigned char *string) size_t _mbstrlen(const char *string) int lstrlen(LPCTSTR lpString)

Examples of Incorrect Code	`char dest[50];`
	`strncpy(dest, src, strlen(src));`

Examples of Corrected Code	`char dest[50];`
	`strncpy(dest, src, sizeof(dest));`
	`dest[sizeof(dest)-1] = 0;`

Source Reference	• http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vclib/html/_crt_strlen.2c_.wcslen.2c_._mbslen.2c_._mbstrlen.asp[2]
Recommended Resources	
Discriminant Set	**Operating System**: • Windows **Languages**: • C • C++

1. mailto:copyright@cigital.com

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.